UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/608,985	06/26/2003	Amitabh Srivastava	3382-64710	6401

26119 7590 03/13/2007

KLARQUIST SPARKMAN LLP
121 S.W. SALMON STREET
SUITE 1600
PORTLAND, OR 97204

EXAMINER	
VU, TUAN A	
ART UNIT	**PAPER NUMBER**
2193	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	03/13/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/608,985	SRIVASTAVA ET AL.	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *18 December 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-36* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-36* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *12/18/06;2/28/07*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 12/18/06.

As indicated in Applicant's response, claims 1, 12-15, 21, 25, 27, 29 have been amended. Claims 1-36 are pending in the office action.

Claim Objections

2. Claims 1 and 27 are objected to because of the following informalities: the phrase recited as 'requesting ... a system dependency creation request ... and a dependency request' appears awkward a language construct or semantic phraseology. It cannot be construed in terms of syntactic norms or semantic level as to how a *requesting* action would be such for requesting both a (system dependency) creation request and a (dependency) request; rendering the meaning of the requesting step obscure even in light of the Specifications (emphasis added), all of which rendering the phrasing of the limitation inappropriate semantically and/or grammatically. For analogy and to show/exemplify how improper a awkward phrase can be perceived,

one would not be *purchasing* a effectuated purchase; or

one would not be determining a determination;

and one would not be requesting a creation request and a dependency request.

The impropriety in language will be treated according to the Examiner's interpretation as set forth in the Examiner's Response to Arguments. Appropriate correction is required.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1-36 are rejected under 35 U.S.C. 102(a) as being anticipated by Srivastava et al., "Effectively Prioritizing Tests in Development Environment", February 2002, MSR-TR-2002-15, Publisher: *Association for Computing Machinery, Inc.*, pp. 1-10 (hereinafter Srivastava).

As per claim 1, Srivastava discloses method comprising:

receiving a system definition (e.g. Fig. 1; *Echelon, by the time ...all changes ... have been propagated* – 5th para, R col. pg. 2; *shifted addresses, different register allocation, small programs modifications* - 4th para, L column, pg. 3; *list of modified and new blocks* – R column, top para, pg. 3 – Note: Echelon infrastructure using Vulcan API as to receiving names of binary files and list of modification of blocks and *changes* to binary headers in terms of shifted addresses or small program changes reads on a application interface receiving system definition about binary files) and a request for dependency information (e.g. BMAT - 4th para, L column, pg. 3; R column, top para, pg. 3);

requesting via an application programming interface, a system dependency creation request comprising the received system definition, and a dependency request comprising a target logical abstraction (e.g. BMAT, API Vulcan, 4th para, L column, pg. 3 – Note: using API in Vulcan to effect a BMAT instance of files mapping result, using a set of binary files – via scanning from Fig. 1 – reads on request comprising a target logical abstraction – see *marked blocks... impacted blocks ...new blocks* – pg. 3, L column, top 2 para); and

receiving responsive to the application programming interface request, a dependency collection for the target logical abstraction comprising logical abstractions in one or more

dependency chains with the target logical abstraction (e.g. Fig. 1, R column pg. 3 – Note: analysis of procedural graph to yield marking information depending on which to effect test coverage reads on abstractions having dependency chains),

wherein the dependency collection comprises logical abstractions outside the target logical abstraction's subsystem (e.g. *marked blocks... impacted blocks ...new blocks* – pg. 3, L column, top 2 para – Note: dependency of successor or predecessor in block analysis via heuristic to prioritized list reads on abstraction within a subsystem and outside a subsystem – see Fig. 2, pg. 4, for iteration per sequence in and out current block);

displaying a representation of the collection (e.g. *impacted Blocks* Table 2-3, pg. 4, 6).

As per claim 2, Srivastava discloses wherein the system definition is received as a file identification (e.g. *header files* - 5th para, R col. pg. 2; Fig. 1).

As per claim 3, Srivastava discloses wherein the system definition is received via a graphical user interface (e.g. *API ...for inspection*, 3rd para, L col. pg. 3).

As per claim 4, Srivastava discloses unchanged logical abstraction (e.g. *old blocks*, 2nd para, R col. pg. 3 – Note: old binary of Fig. 1 in matching reads on unmodified blocks or unchanged logical abstraction – see *logical level ... representation is symbolic*).

As per claims 5-6, Srivastava discloses wherein the dependency collection comprises logical abstractions dependent on the target logical abstraction; wherein the dependency collection comprises logical abstractions on which the target logical abstraction depends (*along with ... coverage information* – L column 1st para, pg. 2; *changes ... propagated* – R column, 5th para, pg. 2; *list of modified blocks* – top para, R column. pg. 3; *marked as old block, impacted blocks, matching blocks* -- pg. 3, R column).

As per claim 7, Srivastava discloses wherein the target logical abstraction is a changed logical abstraction (e.g. *new blocks*, 2nd para, R col. pg. 3 – Note: old binary of Fig. 1 in matching reads on modified blocks or changed logical abstraction).

As per claims 8-11, Srivastava discloses wherein the target logical abstraction comprises a basic block, a procedure, or a binary file; a basic block logical abstraction; one of a procedure logical abstraction or a binary file logical abstraction; a named object logical abstraction or a node logical abstraction (e.g. *procedures, blocks within each ... shifted addresses, different register allocation, small programs modifications* - 4th para, L column, pg. 3; *list of modified and new blocks* – R column, top para, pg. 3; Table 1, pg. 4; *interprocedural graph* -R column,4th para, pg. 3).

As per claim 12, Srivastava discloses representation of the collection comprising a number of affected (e.g. *impacted Blocks* Table 2-3, pg. 4, 6) logical abstractions.

As per claim 13-15, Srivastava discloses representation of the collection comprising a graphical presentation of a dependency chain; a representation of the collection comprising a graph of logical abstractions (e.g. *interprocedural graph* -R column,4th para, pg. 3; Table 2-3, pg. 4, 6).

As per claim 14, Srivastava discloses a list of logical abstractions (e.g. ch. 3.1, list of modified and new blocks – L col. top, pg. 3; Table 4, pg. 7).

As per claim 16, Srivastava discloses logical abstractions inside the logical abstraction's subsystem (e.g. set of sequences, CurrBlkSet – Fig. 2, pg. 4).

As per claims 17-18, Srivastava discloses means for displaying a proposed change risk (e.g. Fig. 4-5, pg. 5 – Note: percentage of impacted blocks over total sequences reads on display

of proposed risk changes in view of proposed changes - *proposed* – top para, R column, pg. 2); i.e. percentage reads on displaying a change risk metric.

As per claim 19, Srivastava discloses wherein the relation is further adapted with a logarithmic function (see Fig. 5, pg. 5).

As per claim 20, Srivastava discloses computer-readable medium having executable instructions for performing the method of claim 1 (refer to claim 1).

As per claim 21, Srivastava discloses method comprising:

determining dependency information about binary files (e.g. *along with ... coverage information* – L column 1st para, pg. 2; *changes ... propagated* – R column, 5th para, pg. 2; *list of modified blocks* – top para, R column. pg. 3; *marked as old block, impacted blocks, matching blocks* -- pg. 3, R column - Note: using previous results or changes in files, and determining blocks for marking and marking insides binary files to find out about subsequent path on how the block test coverage is to be done reads on determining dependency information based on which to apply optimization algorithm upon runtime resources);

propagating dependency information to determine subsystem dependency information for a subsystem of the system; propagating the subsystem dependency information to determine system dependency information for the system (e.g. *changes ... propagated* – R column, 5th para, pg. 2 – Note: marking from using information changes to affected parts of a source binary as this is scanned under analysis reads on propagating this information determination or change information and use it for marking of blocks for a given file version, i.e. subsystem of system or dependency of system – see Table 1, Table 2, pg. 4 in light of algorithm of Fig. 2 where iterative sub-block are included into higher blocks for a coverage);

marking changed logical abstractions; marking unchanged logical abstractions dependent on marked changed logical abstractions in other subsystems (*marked as old block, impacted blocks, matching blocks* -- pg. 3 R column);

comparing test coverage to marked changed logical abstractions and to marked unchanged logical abstractions (e.g. *heuristic ...successor predecessor blocks* – pg. 3, R column ; *likely to be taken* – R column, bottom pg. 3; Fig. 2 – Note: based on marking information between old and new block, and applying heuristic thereto in order to predict how to skip path reads on comparing based on marked of changed and unchanged); and

prioritizing tests based on maximum test coverage of marked changed logical abstractions and marked unchanged logical abstractions (e.g. Fig. 2);

and performing the prioritized tests according to test priorities to produce test results (top para, L col., pg. 2; Figs. 8, 10, pg. 6-7).

As per claims 22-23, Srivastava discloses coverage comprises tests for one subsystem (e.g. sequence Set – Fig 2 – Note: block set being tested based on weight of impacted blocks reads on subsystem of binary blocks), a subsystem among a plural subsystems.

As per claim 24, Srivastava discloses the test coverage comprises tests for plural subsystems and maximum test coverage is considered for marked changed logical abstractions and marked unchanged logical abstractions (pg. 4, L column; Fig. 2 – Note: impacted blocks based on marking of old and new blocks reads on test coverage for marked changed and unchanged – see R col., pg. 3) for said plural subsystems.

As per claim 25, Srivastava discloses a computer-based service comprising **means for** performing the same steps as recited in claim 21, namely:

determining binary dependencies for a defined system (e.g. *along with ... coverage information* – L column 1st para, pg. 2; *changes ... propagated* – R column, 5th para, pg. 2; *marked as old block, impacted blocks, matching blocks* -- pg. 3 R column);

propagating binary dependencies to identify binaries dependent on binaries in other subsystems and storing determined and propagated dependencies (*changes ... propagated* – R column, 5th para, pg. 2 - Note: marking from using information changes to affected parts of a source binary as this is scanned under analysis reads on propagating this information determination or change information and use it for marking);

marking changes and propagating marked changes using the determined and propagated dependencies; and prioritizing tests based on test coverage of marked changes and propagated marked changes;

performing prioritized tests according to the prioritizing;

all of which steps having been addressed in claim 21.

As per claim 26, Srivastava discloses marking proposed changes (e.g. *proposed* – top para, R column, pg. 2; *new version ... likely to be covered by a existing test* – 3rd para, R column, pg. 3 – Note: applying a test to a new code reads on proposed changes to an current test scenario).

As per claim 27, Srivastava discloses a computer-readable medium having executable instructions for performing a method comprising:

creating a system definition in response to receiving graphical user interface input (e.g. *Echelon, by the time ...all changes ... have been propagated* – 5th para, R col. pg. 2; *shifted addresses, different register allocation, small programs modifications* - 4th para, L column, pg. 3;

list of modified and new blocks – R column, top para, pg. 3 – Note: Echelon infracstructure using Vulcan API as to receiving names of binary files – see Fig. 1 – and list of modification of blocks and *changes* to binary headers in terms of shifted addresses or small program changes reads on a application interface receiving system definition about binary files);

receiving a dependency information request via graphical user interface input (e.g. BMAT - 4th para, L column, pg. 3);

requesting via an application programming interface exposed by a dependency framework, a system dependency creation request comprising the system definition, and a target logical abstraction identifiable from the dependency information request (BMAT, API Vulcan, 4th para, L column, pg. 3 – Note: using API in Vulcan to effect a BMAT instance of files mapping result, using a set of binary files – via scanning from Fig. 1 – reads on request comprising a target logical abstraction – see *marked blocks... impacted blocks ...new blocks* – pg. 3, L column, top 2 para); and

receiving responsive to the application programming interface request, a dependency collection for the target logical abstraction comprising logical abstractions in one or more dependency chains with the target logical abstraction (e.g. Fig. 1, R column pg. 3 – Note: analysis of procedural graph to yield marking information depending on which to effect test coverage reads on abstractions having dependency chains);

wherein the dependency collection comprises logical abstractions outside the logical abstraction's subsystem (e.g. *marked blocks... impacted blocks ...new blocks* – pg. 3, L column, top 2 para – Note: dependency of successor or predecessor in block analysis via heuristic to prioritized list reads on abstraction within a subsystem and outside a subsystem – see Fig. 2, pg.

4, for iteration per sequence in and out current block; *interprocedural graph* - R column,4th para, pg. 3).

As per claim 28, Srivastava discloses wherein the dependency collections further comprises logical abstractions inside (Note: dependency of successor or predecessor in block analysis via heuristic to prioritized list reads on abstraction within a subsystem and outside a subsystem – see Fig. 2, pg. 4, for iteration per sequence in and out current block; see *blocks within each procedures* – 4th para, L col., pg. 3) the target logical abstraction's subsystem.

As per claim 29, Srivastava discloses computer system comprising:

a processor coupled to volatile and nonvolatile memory; binary files stored in memory (see Binary - Fig. 1); software stored in memory comprising computer executable instructions for:

determining dependency information for binary files, propagating dependency information to determine subsystem dependency information for a subsystem of a system, and propagating subsystem dependency information to determine system dependency information for the system (refer to claim 21);

marking logical abstractions changed from a previous version (e.g. *Old Binary, New Binary* - Fig .1, R column pg. 3);

propagating marked changes according to the dependency information comprising marking unchanged logical abstractions dependent on marked changes in other subsystems (refer to claim 21);

comparing test coverage to marked changed logical abstractions and to marked unchanged logical abstractions; and prioritizing tests based on maximum test coverage of marked

changed logical abstractions and marked unchanged logical abstractions (refer to corresponding rejection in claim 21).

As per claims 30-31, Srivastava discloses maximum test coverage is based on the total number of marked changed and marked unchanged logical abstractions touched (e.g. Fig. 2; L column, pg. 4) by a test system wide (Note: covering of subset or test sequence per iteration reads on system wide); wherein maximum test coverage is based on the sum of the total number of marked changed logical abstractions in a first subsystem touched by a test, and marked unchanged logical abstractions touched by the test in the first subsystem, wherein the marked unchanged logical abstractions depend on marked changed logical abstractions in other subsystems (e.g. pg. 3, R column 2nd para, bottom para, pg. 3 to L column, pg. 4; ch. 4-5, pg. 4-5).

As per claim 32, Srivastava discloses user interface service (Echelon infracstructure using Vulcan API as to inspecting binary files – see Fig. 1- reads on application with GUI) comprising means for:

accepting a system definition comprising binary files (BMAT – pg. 3, R column, 4th para) in plural subsystems;

accepting an indication (ch. 3.1, pg. 3: ... *list of tests... of modified...blocks* – top R column, pg. 3; *hashing-based ... shifted addresses, different register allocation, program modifications* - pg. 3, L 4th para) of a target logical abstraction;

displaying dependency relationships between the target logical abstraction and a set of logical abstractions in binary files from two or more of the plural subsystems (e.g. Fig. 1; table 1-2, pg. 4; table 3-4, pg. 6-7).

As per claim 33-34, Srivastava discloses means for displaying a proposed change risk (e.g. Fig. 4-5, pg. 5 – Note: percentage of impacted blocks over total sequences reads on display of proposed risk changes in view of proposed changes - *proposed* – top para, R column, pg. 2); i.e. percentage reads on displaying a change risk metric.

As per claim 35, Srivastava discloses displaying a graph of relative risk for plural subsystems (Fig 3-5, pg. 5; Fig. 6-7, Table 3, pg. 6).

As per claim 36, Srivastava discloses means for displaying test coverage evaluation results (see Table 3-4, pg. 6-7).

Response to Arguments

5. Applicant's arguments filed 12/18/06 have been fully considered but they are not persuasive. Following are the Examiner's observations in regard thereto.

35 USC § 102 Rejection under Srivastava:

(A) Applicants have submitted that the Invention provides reception of a system definition describing binary systems in each subsystem; that the recited steps of '*receiving* a system definition ... dependency information; *requesting* via an application ... comprising a target logical abstraction' are provided in Figure 6; which clearly shows a system definition which Srivastava does not have, because Echelon operates directly on binary files, rather than *system definition* files, as set forth in the first recited steps with disclosure from above (Appl. Rmrks, pg. 13; pg. 14, middle). First, the step of *receiving* (a system definition) has been interpreted as being same as (or subsumed in) the step of requesting which also comprises reception of a system definition -- rendering the *receiving* being integral to the *requesting* -- for lack of details about how *receiving* distinguishes over requesting when each encompasses *receiving* of the same

received *system definition*. Second, the claim does not recite 'system definition files' as exemplified via XML of Fig. 6 to particularly characterize 'a system definition' or 'dependency information' as a file or constructs of a markup language of a particular system level meta-format.

Third, the claimed step of *requesting* -- via an API a 'system dependency creation request comprising the received system definition' and a 'dependency request' comprising a target abstraction-- has been interpreted as following: it is via a API that a system *dependency creation request* (Note: this wordy phrase treated as a mere request) is formed; such *creation request* interpreted as constituting of said *system definition* and a dependency construed as *a target logical abstraction*. That is, only 2 entities are fetched from the *creation request*, the system definition and a target logical abstraction. The rejection has profferred Echelon/Vulcan and BMAT as being the means by which those 2 entities are requested and obtained. There is no specificity in describing this recited 'system definition', nor are there further details about 'target logical abstraction'. Broad interpretation has it that such 'system definition' is a representation wherein defined elements can be perceived or visualized (as in Sravistava BMAT using Vulcan) for a global analysis including more refined/definite information concerning the component therein, e.g. programmatic definition such as line number, header, sequence, index in a list, or hashed information, size (see Modification Infrastructure, pg. 3; Table 1, 2, pg. 4). For lack of details about the recited 'definition', this has been interpreted as mere defined elements being organized as a *system dependency* or *chains of dependency* (see claim 1); and when Vulcan (by Srivastava) invokes a tool to fetch the defined blocks for marking their interrelationship in order to assess and to effect their order of execution, the representation of the specific identification of

block as well as subblocks dependency constitutes what is interpreted by the term 'system definition' because without any definition of blocks, there would be impossible to apply any algorithmic iteration (see Srivastava; Fig. 1, 2). Thus, *Echelon/Vulcan* provides a means where static/global abstraction of components in terms of symbolic blocks (with definition of each components via programmatic means) to be analyzed can be displayed for static/dynamic analysis/instrumentation and modification (see Rejection: Fig. 1; *Echelon, by the time ...all changes ... have been propagated* – 5th para, R col. pg. 2; *shifted addresses, different register allocation, small programs modifications* - 4th para, L column, pg. 3; *list of modified and new blocks* – R column, top para, pg. 3 – Note: Echelon infrastructure using Vulcan API as to receiving names of binary files and list of modification of blocks and *changes* to binary headers in terms of shifted addresses or small program changes reads on a application interface receiving system definition about binary files); and by way of a API, which corresponds to the recited 'request', Srivastava's BMAT enables reception of such definition of system components , whereby the components in forms of blocks and relational static information together are received as a combination of system definition and abstraction, the symbolic blocks (Sravistava: pg. 3, R column, top) of a given version of code reading of target abstraction. Therefore, Sravistava has disclosed what constitutes obtaining via a API, request constituting said *system definition* and what is construed as *a target logical abstraction*, as set forth above on the onset. The language of the claim although wordy only amounts to 2 entities, which are considered mapped by the teachings proffered in the Office Action; rendering the above argument not persuasive. Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to

a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

(B) Applicants have submitted that BMAT is a fast tool that matches two versions of binary program; thus cannot teach or suggest 'receiving a system defintion' and 'requesting ... system dependency creation request ... the received system definition' (Appl. Rmrks pg. 15). It is noted that for each portion of the above claimed feature corresponds a teaching that is construed based on Examiner's interpretation; and this interpretation has been set forth above in section A, with clarification as to how the wordiness of the claimed features only amounts to 2 entities being received by an application request process. In order for the claim to read away from what is allegedly referred to as strict focus on analyzing low-level binary construct, it is deemed that *system definition* or *target logical abstraction* or *system dependency creation request* be more defined (in the claim language) in order to enforce a system level description using metadata files, thence distinguishing over the abstract representation of code blocks (previously defined and fetched in the tool) by Srivastava, e.g. to the effect of precluding any interpretation referencing to binary level analysis -- as implied from the argument. The argument is referred back to section A for lack of convincing weight.

(C) Applicants have submitted that the Office Action has reused citations for claim against the 'propagating dependency information ...' and 'propagating subsystem dependency ... for the system'; and as such cannot anticipate claim 21. The rejection of claim 21 has provided distinct citations with respect to addressing the features of claim 1, that is:

(e.g. *along with ... coverage information* – L column 1st para, pg. 2; *changes ... propagated* – R column, 5th para, pg. 2; *list of modified blocks* – top para, R column. pg. 3; *marked as old block, impacted blocks, matching blocks* -- pg. 3, R column - Note: using previous results or changes in files, and determining blocks for marking and marking insides binary files to find out about

subsequent path on how the block test coverage is to be done reads on determining dependency information based on which to apply optimization algorithm upon runtime resources).

Applicants have failed to point out specifically how the cited portions distinguish over the language of the claim; and the Argument amounts to mere allegation for patentability, hence is not persuasive according to 37 CFR § 1.111(b).

(D) Applicants have submitted that the Action by using the similar rejection to address claims 1 and 21 cannot establish that Srivastava discloses or suggests the claims (see claim 25, claim 27, 29, 32 - Appl. Rmrks, pg. 16-17). This form of allegation is referred to section C because sections A and B have addressed the arguments against claim 1; i.e. Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

Double Patenting Rejection:

(E) Applicants have submitted that Srivastava does not teach the *propagating* limitations (Appl. Rmrks pg. 18) therefore, the rejection cannot render '053 claim obvious over Srivastava, lacking proper teaching, suggestion and motivation to combine, thus failing to establish a prima facie case of obiousness Double Patenting. As set forth above, the arguments in regard to Srivastava's not teaching the propagating steps amount to mere allegation that would not characterize as proper prima facie case of a rebut against a rejection (see sections C & D); hence the argument in regard to the propagating steps remain insufficient to overcome the current use of Srivastava in the rejection. The rejection is maintained until proven inappropriate, and in a more convincing manner.

Double Patenting

6. The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. A nonstatutory obviousness-type double patenting rejection is appropriate where the conflicting claims are not identical, but at least one examined application claim is not patentably distinct from the reference claim(s) because the examined application claim is either anticipated by, or would have been obvious over, the reference claim(s). See, e.g., *In re Berg*, 140 F.3d 1428, 46 USPQ2d 1226 (Fed. Cir. 1998); *In re Goodman*, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); *In re Longi*, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); *In re Van Ornum*, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); and *In re Thorington*, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) or 1.321(d) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent either is shown to be commonly owned with this application, or claims an invention made as a result of activities undertaken within the scope of a joint research agreement.

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

7. Claims 21, 25, 29 provisionally rejected on the ground of nonstatutory obviousness-type double patenting as being unpatentable over claims 1, 14 of copending Application No. 11,330,053 (hereinafter '053), further in view of Srivastava et al., "Effectively Prioritizing Tests in Development Environment", February 2002, MSR-TR-2002-15, Publisher: *Association for Computing Machinery, Inc* (hereinafter Srivastava). Although the conflicting claims are not identical, they are not patentably distinct from each other because of the following conflicts.

This is a provisional obviousness-type double patenting rejection because the conflicting claims have not in fact been patented.

As per instant claims 21, 25, and 29, '053 claims 1, 14 also recite prioritizing test (which include performing tests) based on indicators indicating which portions are impacted by tests, portions including plurality of blocks wherein indication (i) indicates ones of the plurality

of blocks are modified, hence this reads on marking changed logical abstractions; but '503 does not recite:

determining dependency information about binary files, propagating such information to determine subsystem and system dependency, marking changed and unchanged logical abstraction to prioritize tests.

Srivastava in a environment for obtaining list prioritized tests (see pg. 5-7) to perform tests, similar to '053; and not only discloses using previous test results or data being marked to feed into new prioritized test, but also discloses binary files analysis to block system and subsystem, recording changes thereto in a propagation analysis hashing (L column, 4th para, pg. 3), and further obtaining dependency information (e.g. *along with ... coverage information* – L column 1st para, pg. 2; *changes ... propagated* – R column, 5th para, pg. 2; *list of modified blocks* – top para, R column. pg. 3; *marked as old block, impacted blocks, matching blocks* -- pg. 3, R column) so as to propagate it into graph elements in order to mark intra-procedural or interprocedural changes so to determine what are new block or unchanged blocks or potential impacted new block in view of a subsequent test coverage (see Fig. 1, Fig. 2, pg. 3-4); and producing prioritized lists (see top pg. 3).

It would have been obvious for one skill in the art to implement the prioritized tests by instant claims 21, 25, and 29 so that '053 steps of creating list of prioritized tests recording modified or new blocks with respect to a given test version, which is analogous to Srivastava from above, be implemented to include using test results into a test optimization instance and effecting for which a dependency information determination and propagation via graph and hashing algorithm (see Srivastava) to enable marking of system and subsystem of abstractions in

binary files as taught by Srivastava, because this enable appropriate marking as suggested in (i) leading to optimization on how to reduce paths in recurring tests for a given test version based on dependency information propagated in target binary files as Srivastava discloses (see Fig. 2).

Conclusion

8. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571)272-3756.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence - please consult Examiner before

using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Tuan A Vu
Patent Examiner,
Art Unit 2193
March 9, 2007